CloudSMS Web API Specification_v2.3_EN

Introduction of Web API

Introduction

This specification is used to illustrate the Web API interface specification of the CloudSMS platform. The CloudSMS platform is a short message service platform developed based on the RESTful style. It uses the common UIP (Unified Interface Platform) standard to regulate communication between computer systems.

Interface specification and communication principle

- Use JSON format to encapsulate UIP interface data. For more information about JSON, please refer to: http://www.json.org/json-zh.html
- Transmission over TCP, the communication port is determined by negotiation between the communication parties.
- Whether you need a long connection can be selected according to the specific service, UIP defines a dedicated heartbeat mechanism to maintain a long connection.
- UIP packets are one-to-one correspondence patterns for requests and responses.
- The following four method fields must be included in each UIP package:
  - METHOD, TYPE, SERIAL, TIME, please refer to the explanation of `General Parameters` for details.
- Both parties to the communication need to use HTTPS for encryption, and SSL for two-way authentication.
- All requests are based on the HTTPS + POST protocol.
- The HTTPS message body is in JSON format.

Noun interpretation and abbreviation

- **LMAX:** Maximum byte length of data;
- **UINT:** Unsigned integer
- **STR:** character string

- **Array:** Array

Version Change Log

Version	Log
CloudSMS 2.0_2021_04_26	• Update CloudSMS environment access address information • Add RESULT_CODE error code description and corresponding solutions • Add LOCAL_REJECTED error code description • Add CloudSMS SMS slicing rule description • New Python request script
CloudSMS 2.1_2021_04_27	• Added bulk dispatching function (SMS_SEND_REQUEST)
CloudSMS 2.2_2021_06_15	• Change the Parameters in DR Callback Method (SMS_PUSH_REPORT)
CloudSMS 2.3_2021_09_13	• Added DCS3 8bit slicing rule

CloudSMS Platform Server

In order to ensure the security of data communication between customers and the CloudSMS platform, the CloudSMS platform will adopt a whitelist restriction method, so as to collect relevant network information of customers as the basis for verification.

	Item	Server link	Remarks
SMS sending request	Test Server	https://cloudsms-new.jegotrip.com.cn:61815/dave/io	Need customers to provide source public network IP address registration whitelist
	Production server	https://cloudsms-new.jegotrip.com.cn:1815/dave/io	Need customers to provide source public network IP address registration whitelist
	Cloud SMS callback	223.119.32.232；223.119.32.134 223.119.32.139；223.119.32.250 223.119.32.160 (STG Test Env)	The sending address belongs to the juxtaposition relationship and will be

SMS status callback	sending server		used randomly. Customers need to join their server whitelist.
	Client callback docking server	Ask customers to submit their docking server link (only one callback address is supported, HTTP/HTTPS link is recommended)	Used to record customer acceptance SMS callback information link.

CloudSMS Business Process

Users can use their customer platform to connect to the CloudSMS platform through the Web API interface (HTTPS). Using the CloudSMS platform, the provided SMS sending function submits an SMS sending request UIP data packet, and the CloudSMS platform sends a response UIP data packet to the customer after receiving the request and verifying success. After the operation is completed, push the status information of the SMS to the customer platform, including short message sending status, sending quantity, cost information, etc.

无法复制加载中的内容

General Parameters

Each UIP packet must contain the following four parameters, which are explained in detail in the following table:

Parameter	Type	LMAX	Description	Required
METHOD	STR	-	Describe the business to which the UIP belongs.	Yes
TYPE	STR	8	Represents the type of the UIP: request or response	Yes
SERIAL	UINT	4	Used to record the transmission sequence number of UIP packets.	Yes
TIME	STR	32	Record the local time the UIP packet was issued.	Yes

METHOD

Used to describe the service to which the UIP belongs, providing two choices:

- **SMS_SEND_REQUEST:** means requesting SMS sending operation;
- **SMS_PUSH_REPORT:** Indicates the request SMS status report operation.

TYPE

Represents the type of the UIP, providing two value choices:

- **REQUEST:** Represents the type of the requested UIP message;
- **ANSWER:** Represents the type of response UIP message.

SERIAL

Used to regulate the order of UIP packet transmission, counted by each of the transmitting parties, starting from 1 to 128 for cyclic counting. The role of this field is mainly used for fault tolerance, in order to prevent the existence of data packets are not processed or abnormal system. (greater than or equal to 1, less than or equal to 65535)

TIME

Used to record the local time when a UIP packet was issued.

- For example: Beijing time (UTC + 8), the time record format is: YYYY-MM-DD HH: MM: SS

How to use CloudSMS platform

CloudSMS platform supports two short message request functions, namely short message sending request and short message status report push. Each request UIP data packet needs to have a corresponding response packet corresponding to it.

SMS Dispatching

Introduction

When the method name is set to **SMS_SEND_REQUEST**, it means that a short message is sent through this UIP packet request. Customers can add custom parameters in JSON format to the request packet. Custom parameters will be fed back in the callback DR. The request UIP packet details are shown in the following table.

Request UIP Format

Method	POST

Parameter	Type	LMAX	Description	Required
METHOD	STR	-	SMS_SEND_REQUEST	Yes
TYPE	STR	8	REQUEST	Yes
SERIAL	UINT	4	Value 1~ 65535	Yes
TIME	STR	32	Record the local time the UIP packet was issued.	Yes
AUTH_KEY	STR	128	Fixed character string length of 128 bytes.	Yes
MULTI_MSISDN_LIST	Array		Used to carry phone number data. Supports up to 200 numbers at a time.	Yes
MULTI_MSISDN_LIST array Start				
COUNTRY_CODE	UINT	4	Used to carry country and area codes, [Chinese mainland: 86]	Yes
DEST_MSISDN	STR	32	Phone number for carrying purpose phone	Yes
MULTI_MSISDN_LIST array End				
SMS_CONTENT	STR	1600	This field carries the specific content of SMS	Yes
ROUTE_ID	STR	255	Represents the routing policy name for the CloudSMS platform.	Yes

			Generated by relevant operation and maintenance personnel configuration specific policies.	
ORIGINAL_ADDR	STR	32	A number or letter used to carry an SMS number	No
SIGNATURE	STR	64	Signed character string with SMS	No
SIGNATURE_TYPE	UINT	4	Set the format and position of the signature	No
PRIORITY	UINT	4	Used to set the priority of short messages. ◦ 0: general, pass priority (default scheme) ◦ 1: High priority ◦ > 1: Reserved	No
VERSION	STR	12	Represents the current version of CloudSMS Value: 2021-01-01	Yes
CUSTOMER_BODY	OBJ	255	Store customer defined JSON structure data	No

Response UIP Format

Each UIP request packet requires one-to-one correspondence with a response packet, as shown in the following table.

Parameter	Type	LMAX	Description	Required
METHOD	STR	-	SMS_SEND_REQUEST	Yes
TYPE	STR	8	ANSWER	Yes
SERIAL	UINT	4	Value 1~ 65535	Yes
TIME	STR	32	Record the local time the UIP packet was issued.	Yes

RESULT_CODE	INT	4	Result code reply	Yes
RESULT_DESC	STR	127	Returns the descriptive information of the result code A displayable character in the ANSI character set.	Yes
DETAIL_LIST	Array	--		Yes
DETAIL_LIST array Start				
COUNTRY_CODE	UINT	4	This field is used to carry the country area code. [Chinese mainland: 86]	Yes
DEST_MSISDN	STR	32	The telephone number of the short message destination telephone.	Yes
SMS_UID	STR	32	Number character string within 32 bits, as the unique identification of SMS	Yes
RESULT_CODE	INT	4	Result code reply	Yes
RESULT_DESC	STR	127	Returns the description information of the result code	Yes
DETAIL_LIST array End				

Remark: Support small batch SMS in `MULTI_MSISDN_LIST` field, maximum support 200. Each SMS will be assigned a `SMS_UID`.

Examples

Example 1 - Single SMS

SMS Request (Single SMS)

Python

```
{
    "METHOD": "SMS_SEND_REQUEST",
    "TYPE": "REQUEST",
    "SERIAL": 1,
    "TIME": "Please input your local time",
    "AUTH_KEY": "Please input your auth_key here",
    "ROUTE_ID": "Please input your Route_ID here",
    "MULTI_MSISDN_LIST":
    [
        {
            "DEST_MSISDN": "67xxxx11",
            "COUNTRY_CODE": 852
        }
    ],
    "SMS_CONTENT": "Your verification code is 123345, please do not disclose it to others.",
    "SIGNATURE": "CloudSMS", # The SMS signature does not need to be filled in [];
    "SIGNATURE_TYPE": 1, # If you use the SIGNATURE parameter, select the SIGNATURE_TYPE field
    "ORIGINAL_ADDR":"", # If the parameter value is left unfilled or empty, the SMS will use the default SenderID
    "VERSION": "2021-01-01",
    "REPEAT":0,
    "CUSTOMER_BODY":{}
}
```

SMS Response (Single SMS)

Python

```
{
    "METHOD": "SMS_SEND_REQUEST",
    "TYPE": "ANSWER",
    "SERIAL": 1,
    "TIME": "System Response Time", # Format: YYYY-MM-DD HH:MM:SS
    "RESULT_CODE": 1, # Result_code Refer to the Appendix for the RESULT_CODE and RESULT_DESC parameters for details
    "RESULT_DESC": "'OK'",
    "DETAIL_LIST":
    [
        {
            "RESULT_CODE": 1,
            "RESULT_DESC": "'OK'",
            "DEST_MSISDN": "67xxxx11",
            "COUNTRY_CODE": 852,
            "SMS_UID": "5A3B394096E50X01D1FA163E81BFFF"
        }
    ]
}
```

Example 2 - Bulk SMS

SMS request (Bulk SMS)

Python

```
{
    "METHOD": "SMS_SEND_REQUEST",
    "TYPE": "REQUEST",
    "SERIAL": 1,
    "TIME": "Please input your local time",
    "AUTH_KEY": "Please input your auth_key here",
    "ROUTE_ID": "Please input your Route_ID here",
    "MULTI_MSISDN_LIST":
    [
        {
            "DEST_MSISDN": "67xxxx11",
            "COUNTRY_CODE": 852
        },
        {
            "DEST_MSISDN": "67xxxx28",
            "COUNTRY_CODE": 852
        }
    ],
    "SMS_CONTENT": "Your verification code is 123345, please do not disclose it to others.",
    "SIGNATURE": "CloudSMS", # The SMS signature does not need to be filled in []; the box will be added via SIGNATURE_TYPE
    "SIGNATURE_TYPE": 1, # If you use the SIGNATURE parameter, select the SIGNATURE_TYPE field
    "ORIGINAL_ADDR":"", # If the parameter value is left unfilled or empty, the SMS will use the default SenderID
    "VERSION": "2021-01-01",
    "REPEAT":0,
    "CUSTOMER_BODY":{}
}
```

SMS response (Bulk SMS)

Python

```
{
    "METHOD": "SMS_SEND_REQUEST",
    "TYPE": "ANSWER",
    "SERIAL": 1,
    "TIME": "响应时间", # Format: YYYY-MM-DD HH:MM:SS
    "RESULT_CODE": 1, # Result_code Refer to the Appendix for the RESULT_CODE and RESULT_DESC parameters for details
    "RESULT_DESC": "'OK'",
    "DETAIL_LIST":
    [
        {
            "RESULT_CODE": 1,
            "RESULT_DESC": "'OK'",
            "DEST_MSISDN": "67xxxx11",
            "COUNTRY_CODE": 852,
            "SMS_UID": "5A3B394096E50X01D1FA163E81BFFF"
        },
        {
            "RESULT_CODE": 1,
            "RESULT_DESC": "'OK'",
            "DEST_MSISDN": "67xxxx28",
            "COUNTRY_CODE": 852,
            "SMS_UID": "5A3B394096E50X02D1FA163E81BCCC"
        }
    ]
}
```

Delivery Report (DR) CallBack

CloudSMS supports the operation of pushing DR receipt to customer DR system. The customer needs to deploy an HTTP/HTTPS server to receive DR receipt information pushed from CloudSMS platform.

Remark: The maximum return time of DR receipt is 24 hours. If CloudSMS does not receive the DR report returned by the landing operator after more than 25 hours, CloudSMS will consider the SMS as a timeout state and reply to the DR receipt of `EXPIRED` status to the customer DR system.

Introduction

When the METHOD parameter is set to `SMS_PUSH_REPORT`, it means that the CloudSMS platform will send a status report to the customer SMS DR server through the POST method. The UIP packet details are shown in the following table.

Delivery Report (DR) Request Format

Request method	POST

Parameter	Type	LMAX	Description	Required
METHOD	STR	--	SMS_PUSH_REPORT	Yes
TYPE	STR	8	REQUEST	Yes
SERIAL	UINT	4	Value 1~ 65535	Yes
TIME	STR	32	UTC + 8 Time (Hong Kong Time)	Yes
SMS_UID	STR	32	Unique ID of SMS	Yes
SOURCE_MSISDN	STR	32	Phone number with SMS source	Yes
SMS_RESULT_LIST	Array	--	--	Yes
SMS_RESULT_LIST array Start				
COUNTRY_CODE	UINT	4	Used to carry country and area codes, [Chinese mainland: 86]	Yes
DEST_MSISDN	STR	32	Phone number carrying text message target	Yes
SMS _ STATE	UINT	4	The final state of the text message	Yes
RESULT_DESC	STR	32	Explanation of the SMS report	Yes
SMS_RESULT_LIST array End				

REPORT_TIME	STR	32	Report submission time, UTC + 8 time (Hong Kong time)	No
CUSTOMER_BODY	OBJ	255	Carry customer custom JSON structure data	No

Delivery Report (DR) response Format

Parameter	Type	LMAX	Description	Required
METHOD	STR	-	SMS_PUSH_REPORT	Yes
TYPE	STR	8	ANSWER	Yes
SERIAL	UINT	4	Value 1~ 65535	Yes
TIME	STR	32	Record the local time when the UIP packet was issued	Yes
RESULT_CODE	INT	4	Result code reply	Yes
RESULT_DESC	STR	127	Returns the result code description information	Yes

Examples

DR callback request Sample (Request by CloudSMS)

If you use the bulk SMS group function, the DR receipt will still be pushed to the customer DR system separately according to the SMS_UID.

Python

```
  {
    "METHOD": "SMS_PUSH_REPORT",
    "TYPE": "REQUEST",
    "SERIAL": 2,
    "TIME": "DR Push Time", # Format: YYYY-MM-DD HH:MM:SS
    "REPORT_TIME": "DR Submit Time", # Format: YYYY-MM-DD HH:MM:SS
    "SOURCE_MSISDN": "CloudSMS",
    "SMS_UID":"5A3B394096E50X01D1FA163E81BFFF",
    "SMS_RESULT_LIST":
    [
        {
            "COUNTRY_CODE": 852,
            "DEST_MSISDN": "67xxxx11",
            "SMS_STATE": 2,
            "RESULT_DESC": "DELIVERED",
        }
    ],
    "CUSTOMER_BODY":{}
}
```

DR callback response Sample

Python

```
{
    "METHOD": " SMS_PUSH_REPORT",
    "TYPE": "ANSWER",
    "SERIAL": 2,
    "TIME": "Response Time", # Format: YYYY-MM-DD HH:MM:SS
    "RESULT_CODE": 1,
    "RESULT_DESC": "OK"
}
```

Parameter Details

AUTH_KEY

This parameter is a character string with a fixed length of 128 bytes. It is used to carry an identity so that the CloudSMS server can identify the user through this parameter.

`For example:`

`"AUTH_KEY": "cOUlfIimjcBgfxf5cNaiOZ7BLsonagmjt5+vncIGdKOsZXsjkuQjERKtSrm7iTvNKgcXvvLsK9+ptfYQHFgB0a=="`

SMS_CONTENT

This parameter is used to carry SMS content and allows a maximum length of 1530 bytes. The byte calculation rules are as follows:

- In English (punctuation marks are all half corners), one character and one byte are used.
- The rest is calculated as one character and two bytes (mixed Chinese and English, and English is also two bytes).
- According to the SMPP protocol, long SMS will be split and billed by the CloudSMS platform. For details of the slicing rules, please refer to the `Appendix CloudSMS SMPP slicing rules`

ROUTE_ID

This parameter represents the routing name of the CloudSMS platform, which is automatically generated by the CloudSMS routing policy.

For example: `"ROUTE_ID": "VerifyCode_516"`

SIGNATURE

For directions that have regulatory requirements (example mainland China), it is necessary to successfully register SMS signature on the CloudSMS platform in advance, such as [CloudSMS]. To apply for SMS signature, please contact the sales manager.

SIGNATURE_TYPE

The style used to format the signature and the display position in the SMS. The details are shown in the following table:

SIGNATURE_TYPE	Type	Location	Description
1	Solid square brackets []	At the beginning of the text message	【Signature】xxx…
2	Solid square brackets []	End of SMS	xxx…【Signature】
3	Square brackets []	At the beginning of the text message	[Signature] xxx…
4	Square brackets []	End of SMS	xxx…[signature]
5	None	None	Signature has been spliced, no signature position is set

CUSTOMER_BODY

`CUSTOMER_BODY` parameter is a field that carries the customer defined parameters. This parameter will be pushed to the customer's DR system without changes in the Delivery Report.

For example, customers can customize the identification codes for each short message, and obtain identification codes in DR receipt for short message identification of customers' internal systems.

RESULT_CODE and RESULT_DESC

The result code is used in the response to indicate the acceptance result of the requested UIP packet.

RESULT_CODE	RESULT_DESC	Description	Solution
0x01	OK	Success.	
0x02	Unknown_error	Unknown exception error.	Please report to CloudSMS
0x03	Busy	The platform is busy.	Please try again later (10 minutes)

0x04	Empty_data	Empty packet.	Please check that the API request packet is submitted correctly
0x05	Channel_not_exist	AUTH_KEY mistakes.	Please check whether the AUTH_KEY is filled in correctly
0x10000001	Invalide_paramete	Parameter error.	Check that the API request package parameters are correct. *For example: whether the Parameter is correct, whether the Type is correct*
0x10000003	Invalid_context	Invalid SMS content, SMS content is lost or too long.	Check whether the API packet is in JSON format; Check that the API packet required parameters are complete.
0x10000004	Send_failed	SMS sending failed.	Please report to CloudSMS
0x10000005	Queue_too_large	The cache is full.	Please try again later (10 minutes)
0x10000006	Intercept	Text messages are intercepted (possibly blacklisted).	Please try again later (10 minutes)
0x10000007	Lost_info	Incomplete packet (partial loss)	Check that the API packet required parameters are complete.
0x1000000D	Empty_data	Received empty packet	Please check that the customer API packet format and required parameters are complete.
0x1000000E	DB_not_find	Data storage error	Please try again later (10 minutes)
0x1000000F	Unpriced_area	Prices are not allowed to be sent or set in this area	Please check the SMS direction that supports sending.
0x10000010	Invalid_Version	Invalid version	Please change to the latest Web API version number.

Remark

`Invalid_parameter (0x10000001)` error code will appear in the following cases, please refer to:

- Required parameters missing or not filled in
- The character value of the `SOURCE_MSISDN` is not in the range of ASSIC code `0x20-0x80`.
- Invalid country code
- The value of the `DEST_MSISDN` parameter is non-numeric or longer than 32 bytes
- The number sent to mainland China is not 11 digits in length

SMS_STATE and RESULT_DESC

The status code is used to indicate the transmission state of the short message in the platform short message status push operation.

SMS _ STATE	RESULT_DESC	Description
1	ENROUTE	Message is transmitting
2	DELIVERED	Message has arrived
3	EXPIRED	Message timeout
4	DELETED	Message deleted
5	UNDELIVERABLE	Message cannot be delivered
6	ACCEPTED	The message has been received (for example, it has been manually read by customer service on behalf of the user)
7	UNKNOWN	Invalid message status
8	REJECTED	Message rejected
9	IDLE	Message is being processed
60	LOCAL_REJECTED	SMS is rejected locally by CloudSMS platform

Remark

An error code `LOCAL_REJECTED (60)` is generated in the following cases:

- Singapore Number Validation Rules:
  - Singapore phone numbers don't start with 8 or 9.
  - Singapore phone number is not eight digits (excluding country code)
- Unconfigured sending direction (not included in the product details)
- Wrong `ROUTE_ID` parameter values used in `SMS_SEND_REQUEST`
- Insufficient account balance

Appendix

CloudSMS SMPP slicing rules

Standard English Characters (GSM DCS0 7bit encoding)	**Segment Quantity(s)**
1 – 160 characters	1
161 – 306 characters	2
307 – 459 characters	3
460 – 612 characters	4
613 – 765 characters	5
766 – 918 characters	6
919 – 1071 characters	7
1072 – 1224 characters	8
1225 – 1377 characters	9
1378 – 1530 characters	10

Latin 1 (ISO-8859-1) (DSC3 8bit encoding)	**Segment Quantity(s)**
1 – 140 characters	1

141 - 268 characters	2
269 - 402 characters	3
403 - 536 characters	4
537 - 670 characters	5
671 - 804 characters	6
805 - 938 characters	7
938 - 1072 characters	8
1073 - 1206 characters	9
1207 - 1340 characters	10

Non-GSM (Unicode) characters (DCS8 16bit encoding)	**Segment Quantity(s)**
1 – 70 characters	1
71 – 134 characters	2
135 – 201 characters	3
202 – 268 characters	4
269 – 335 characters	5
336 – 402 characters	6
403 – 469 characters	7
470 – 536 characters	8
537 – 603 characters	9
604 – 670 characters	10

General Format of OTP SMS

Verify Code	[CloudSMS] Your Verification Code: 123456. Please do not disclose this code to anyone. Anyone request for this code is recognized as a fraud. Please do not include this code in any short message or email.

Support settlement currency

Currency name	Currency abbreviation
人民币	CNY
港币	HKD
台币	TWD
欧元	EUR
美元	USD
英镑	GBP
澳元	AUD
韩元	KRW
日元	JPY

Sample Code

Python

Python

```python
import tornado.web
import tornado.ioloop
import requests
import time
import json

```

```python
local_time = time.strftime("%Y-%m-%d %H:%M:%S", time.localtime())

# CloudSMS Server Address
url = ""

data = {
    "METHOD": "SMS_SEND_REQUEST",
    "TYPE": "REQUEST",
    "SERIAL": 1,
    "TIME": local_time,
    "AUTH_KEY": "Please input your AUTH_KEY",
    "MULTI_MSISDN_LIST":
    [
        {
            "DEST_MSISDN": "67xxxx11",  # Phone Number
            "COUNTRY_CODE": 852,  # Country Code
        }
    ],
    "SMS_CONTENT": "Please input your sms content",
    "ROUTE_ID": "Please input the ROUTE_ID",
    "PRIORITY": 0,
    "SIGNATURE": "CloudSMS",  # If sending to mainland China or overseas where t
here is a regulatory need, you need to apply to CloudSMS in advance
    "SIGNATURE_TYPE": 3,
    "ORIGINAL_ADDR": "CloudSMS", # Not required. If it is not filled in or the v
alue is empty, the default value Cloud is usedSMS
    "VERSION": "2021-01-01",  # Please use the latest version
}

data = json.dumps(data)
print("Json: ", data)

class IndexHandler(tornado.web.RequestHandler):
    def post(self):
        jsonbyte = self.request.body
        jsonstr = jsonbyte.decode('utf-8')
        print(jsonstr)

rsp = requests.post(url, data, verify=False).content.decode('utf-8')
print("rsp: ", rsp)


if __name__ == "__main__":
    app = tornado.web.Application(
        [(r'/', IndexHandler)], static_path='static', debug=True, autoreload=Fal
```

```
    se)
50      app.listen(59999)
51      tornado.ioloop.IOLoop.current().start()
```

Ruby

Ruby

```ruby
require 'net/http'
require 'json'

def send_post(url,toSend)
  uri = URI(url)
  http = Net::HTTP.new(uri.host, uri.port)
    if uri.scheme == "https"
      http.use_ssl = true
        http.verify_mode = OpenSSL::SSL::VERIFY_NONE
    end

  req = Net::HTTP::Post.new(uri.path, initheader = {'Content-Type' =>'application/json'})
  req.body = toSend
  res = http.request(req)
  return JSON.parse res.body
end

data = {
  "METHOD": "SMS_SEND_REQUEST",
  "TYPE": "REQUEST",
  "SERIAL": 1,
  "TIME": "Please input your time", # Format: YYYY-MM-DD HH:MM:SS
  "AUTH_KEY": "Please input your AUTH_KEY”,
  "MULTI_MSISDN_LIST":
  [
      {
          "DEST_MSISDN": "67xxxx11",
          "COUNTRY_CODE": 852,
      }
  ],
  "SMS_CONTENT": "Please input your sms content",
  "ROUTE_ID": "Please input the ROUTE_ID",
  "PRIORITY": 0,
  "SIGNATURE": "CloudSMS",
  "SIGNATURE_TYPE": 3,
  "VERSION": "2021-01-01",
  }.to_json

tester = send_post("CloudSMS Server", data)
puts tester
```

Node.js

JavaScript

```javascript
const https = require("https");

const data = JSON.stringify({
  METHOD: "SMS_SEND_REQUEST",
  TYPE: "REQUEST",
  SERIAL: 1,
  TIME: "Please input your time", # Format: YYYY-MM-DD HH:MM:SS
  AUTH_KEY:  "Please input your AUTH_KEY",
  MULTI_MSISDN_LIST: [
    {
      DEST_MSISDN: "67xxxx11",
      COUNTRY_CODE: 852,
    },
  ],
  SMS_CONTENT: "Please input your sms content",
  ROUTE_ID: "Please input your AUTH_KEY",
  PRIORITY: 0,
  SIGNATURE: "CloudSMS",
  SIGNATURE_TYPE: 3,
  VERSION: "2021-01-01",
});

const options = {
  hostname: "",
  port: 1815,
  path: "/dave/io",
  method: "POST",
  headers: {
    "Content-Type": "application/json",
    "Content-Length": data.length,
  },
};

const req = https.request(options, (res) => {
  console.log(`statusCode: ${res.statusCode}`);

  res.on("data", (d) => {
    process.stdout.write(d);
  });
```

```
});

req.on("error", (error) => {
  console.error(error);
});

req.write(data);
req.end();
```

Java – RestTemplate

Python

```
package com.example.demo;

import org.json.JSONObject;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.http.HttpEntity;
import org.springframework.http.HttpHeaders;
import org.springframework.http.MediaType;
import org.springframework.http.ResponseEntity;
import org.springframework.web.client.RestTemplate;

@SpringBootApplication
public class DemoApplication {

    public static void cloudSmsPost() {

        RestTemplate template = new RestTemplate();
        // Request URL
        String url = ""; # CloudSMS Server Address

        // JSON Data
        JSONObject msisdn_list = new JSONObject();
        msisdn_list.put("DEST_MSISDN", "67xxxx11");
        msisdn_list.put("COUNTRY_CODE", 852);

        JSONObject param = new JSONObject();
        param.put("METHOD", "SMS_SEND_REQUEST");
        param.put("TYPE", "REQUEST");
        param.put("SERIAL", 1);
```

```
30          param.put("TIME", "Please input your time");
31          param.put("AUTH_KEY", "Please input your AUTH_KEY");
32          param.put("SMS_CONTENT", "Please input your sms content");
33          param.put("ROUTE_ID", "Please input your AUTH_KEY");
34          param.put("ORIGINAL_ADDR", "CloudSMS");
35          param.put("PRIORITY", 0);
36          param.put("SIGNATURE", "CloudSMS");
37          param.put("SIGNATURE_TYPE", 3);
38          param.put("VERSION", "2021-01-01");
39          param.put("MULTI_MSISDN_LIST", msisdn_list);
40          // POST Headers
41          HttpHeaders headers = new HttpHeaders();
42          headers.setContentType(MediaType.APPLICATION_JSON);
43          HttpEntity<String> entity = new HttpEntity<String>(headers);
44          ResponseEntity<String> result = template.postForEntity(url, entity, Stri
    ng.class);
45
46          // POST
47          String body = entity.getBody(); // get respond body
48          System.out.println("Param: " + param.toString());
49          System.out.println("Respond_body: " + body);
50          System.out.println("Respond_result: " + result);
51      }
52
53      public static void main(String[] args) {
54          SpringApplication.run(DemoApplication.class, args);
55          DemoApplication.cloudSmsPost ();
56      }
57
58  }
```